# 语义分析——符号表信息登录

## 一、实验目的

1. 了解语义分析的主要方法和功能，能判断语法制导定义类型，并根据不同类型采用不同的语义分析方法。

2. 掌握利用语义规则进行符号表信息登录的功能。

## 二、实验内容

针对下表的翻译模式，编写一个 语义分析程序，实现翻译以下句子。

测试句子： id1 : real ; id2 :↑integer; id3:integer

|  |  |
| --- | --- |
| 产生式规则 | 语义规则 |
| P—>MD | { addwidth(top(tblptr),top(offset));  pop(tblptr); pop(offset)} |
| M—>ε | { t = mktable(nil); push(t,tblptr); push(0,offset)} |
| D—>D1; D2 |  |
| D—>proc id; N D1; S | { t = top(tblptr); addwidth(t,top(offset)); pop(tblptr); pop(offset); enterproc(top(tblptr), id.name, t)} |
| D—>id:T | { enter(top(tblptr),id.name,T.type,top(offset));  top(offset) = top(offset) + T.width} |
| N—>ε | { t = mktable(top(tblptr));  push(t,tblptr); push(0,offset)} |
| T—>integer | {T.type = integer; T.width = 4} |
| T—>real | {T.type = real; T.width = 8} |
| T—>↑T1 | {T.type = pointer(T1.type); T.width = 4} |

说明：

1. mktable(previous):创建一张新的符号表，并返回指向新表的指针。参数previous指向先前创建的符号，放在新符号表的表头。
2. enter(table, name, type, offset)：在table指向的符号表中为名字name建立新表项，同时将类型type及相对地址offset放入该表项的属性域中。
3. addwidth(table, width)：将table指向的符号表中所有表项的宽度之和记录在与符号表关联的表头中。
4. enterproc(table, name, newtable)：在table指向的符号表中为过程name建立一个新表项，参数newtable指向过程name的符号表。
5. 栈tblptr：保存指向外围过程符号表的指针。
6. 栈offset：其栈顶元素是下一个当前过程中局部对象可用的相对地址。

## 三、实验要求

1. 在编程前，根据上述翻译模式，确定要采用的计算方法。

2. 输出格式：符号表的表项（含属性值），每条单独一行。

3. 任选 熟悉的编程语言，要求所编程序结构清晰。

4. 完成实验报告及总结。